

IN THE UNITED STATES DISTRICT COURT
FOR THE NORTHERN DISTRICT OF CALIFORNIA

ORACLE AMERICA, INC.,

Plaintiff,

No. C 10-03561 WHA

v.

GOOGLE INC.,

Defendant.

ORDER DENYING RULE 50 MOTIONS

In this copyright case, the Federal Circuit remanded for a second jury trial on the issue of fair use, rejecting the argument of Oracle America, Inc., that the first trial record entitled it to judgment as a matter of law and that a remand on that issue would be “pointless” (Br. at 68). Now, after an adverse verdict in the second trial, Oracle again asserts that it is entitled to judgment as a matter of law on fair use. For the same reasons as before, Oracle is wrong in saying that no reasonable jury could find against it.

Under the law as stated in the final charge and on our trial record, our jury could reasonably have found for either side on the fair use issue. Our trial presented a series of credibility calls for our jury. Both sides are wrong in saying that all reasonable balancings of the statutory factors favor their side only. To the extent either side now quarrels with the law as stated in the final charge (Dkt. No. 1950), the time for those arguments was at or before the charging conference or eventually on appeal. For now, at the district court, the jury instructions control.

1 Based on those instructions, the Rule 50 motions must be **DENIED**. Since an appeal
2 is promised, however, it may be of assistance to leave a few important observations.

3 1. The fair use instructions followed largely the review of fair use law as set forth
4 in the Federal Circuit's opinion except for modifications urged by counsel and to account for
5 how the case was actually tried. The final jury charge culminated an exhaustive and iterative
6 process of proposals by the judge followed by critiques by counsel. Months before trial, the
7 Court informed both sides that it expected to use the Federal Circuit's opinion canvassing
8 fair use law as the starting point and requested briefing from the parties addressing what
9 modifications should be made (Dkt. Nos. 1518, 1519 at 51). After reviewing those comments,
10 the Court circulated a first proposed charge on fair use and requested critiques (Dkt. No. 1615).
11 Counsel submitted their critiques a week later with replies the following week. In light of the
12 critiques, a second draft made substantial revisions (Dkt. Nos. 1688, 1716), asking counsel to
13 meet and confer to reach an agreed-on instruction in light of that proposal and to submit briefs
14 and responses regarding the areas of disagreement. After reviewing the further briefs and
15 responses, the Court next circulated "penultimate instructions on fair use," a third draft, and
16 invited a third round of comment (Dkt. No. 1790). Those critiques also led to modifications
17 and a final notice of the pre-instruction on fair use to be read to the jury before the start of the
18 evidence (Dkt. No. 1828). Counsel (and the jury) were advised that the final instructions at the
19 end of the evidence would possibly be adjusted to reflect the way the case was tried (and, in
20 fact, some minor modifications did occur). During the trial, the judge sought briefs on several
21 issues in play as the evidence came in. Based thereon, a notice of the proposed final charge
22 circulated the night before the close of evidence (Dkt. No. 1923). At the charging conference,
23 counsel raised both new points and old ones (although they were permitted to rest on prior
24 critiques). Final modifications followed. The jury was charged accordingly (Dkt. No. 1950).

25 2. On fair use, Oracle's most emphatic argument remains the "propriety of the
26 defendant's conduct," meaning the subjective awareness by Google Inc. of the copyrights
27 and, construing its internal e-mails in a light most unfavorable to Google, its "bad faith."
28 This, however, underscores an important point for the appeal. Although the Federal Circuit

opinion omitted any reference to the “propriety of the defendant’s conduct” (good faith versus bad faith) as a consideration under any part of the four-factor test for fair use, Oracle insisted on remand that the jury be told that it could consider bad faith by our accused infringer as a subfactor under Factor One. This Court acquiesced in Oracle’s view and did so despite an omission — conceivably a studied omission — of any such consideration in the Federal Circuit opinion and despite the fact that there is a respectable view that good or bad faith should no longer be a consideration after the Supreme Court’s decision in *Campbell v. Acuff-Rose Music, Inc.*, 510 U.S. 569, 585 n.18 (1994). See 2 Paul Goldstein, *Goldstein on Copyright* § 12.2.2, at 12:44.5–12:45 (3d ed. 2016). Put differently, either a use is objectively fair or it is not and subjective worry over the issue arguably should not penalize the user.¹

Still, Oracle is correct that the Supreme Court in *Harper & Row Publishers, Inc. v. Nation Enterprises*, 471 U.S. 539, 562 (1985), called out the propriety of the defendant’s conduct as a consideration in that case. Footnote 18 in *Campbell* later questioned whether or not propriety should persist as a consideration but did not rule it out. At the district court level, we must treat *Harper & Row* as still the law and leave it to the appellate courts to revise (although our instructions included a modification based on *Campbell*). This is no small point in this case, for no Oracle jury argument received more airtime than its argument that Google “knew” it needed a license and chose in bad faith to “make enemies” instead.

This leads to a reciprocal key point. Given that Oracle was allowed to try to prove Google acted in bad faith, Google was allowed to try to prove good faith. Its witnesses testified

¹ On appeal, Oracle argued as follows (Br. 72):

Finally, “[f]air use presupposes good faith and fair dealing.” *Harper & Row*, 471 U.S. at 562 (internal quotation marks omitted). Google considered, negotiated, and ultimately rejected the opportunity to license the packages, deciding to “[d]o Java anyway and defend our decision, perhaps making enemies along the way.” A1166. That Google knew it needed a license, and then sought but did not obtain one, weighs heavily in showing “the character of the use” was not fair. *Los Angeles News Serv. v. KCAL-TV Channel 9*, 108 F.3d 1119, 1122 (9th Cir. 1997). Google “knowingly . . . exploited a purloined work for free that could have [otherwise] been obtained.” *Id.*

Despite this argument, the Federal Circuit did *not* include this consideration in its discussion of factors on fair use, remaining silent on the issue.

1 that they had understood that “re-implementing” an API library was a legitimate, recognized
2 practice so long as all that was duplicated was the “declaring code” and so long as the
3 duplicator supplied its own “implementing code,” that is, the methods were “re-implemented.”
4 In this way, Java programmers using the Android API could call on functionalities with the
5 same Java command statements needed to call the same functionalities in the Java API, thereby
6 avoiding splintering of the ways that identical functionalities became invoked by Java
7 programmers.

8 Google asked to go a step further and asked for an instruction on “custom,” citing
9 *Wall Data Inc. v. Los Angeles County Sheriff’s Dept.*, 447 F.3d 769, 778 (9th Cir. 2006),
10 which stated “fair use is appropriate where a ‘reasonable copyright owner’ would have
11 consented to the use, *i.e.*, where the ‘custom or public policy’ at the time would have defined
12 the use as reasonable” (quoting Subcomm. on Patents, Trademarks & Copyrights of the Sen.
13 Comm. on the Judiciary, 86th Cong., 2d Sess., Study No. 14, *Fair Use of Copyrighted Works* 15
14 (Latman) (Comm. Print 1960)). Oracle objected on the ground that custom was omitted from
15 the Federal Circuit opinion. It was omitted from that opinion — true. But neither was there
16 any mention in that opinion of the propriety of the accused infringer’s conduct. So, that
17 omission by itself wasn’t a good reason to ignore a pertinent statement by the Ninth Circuit, the
18 law applicable in this copyright case arising in the Ninth Circuit. Oracle also argued that
19 “custom” had to be vastly more entrenched than the “practice” evidence Google wished to
20 present.

21 Whether or not the evidence would have warranted a *Wall Data* instruction on custom,
22 the fact remained that once Oracle endeavored to prove bad faith, it opened the door for Google
23 to prove good faith, so Google explained its mental state and explained that it believed it had
24 followed a recognized practice in freely re-implementing API libraries by duplicating only
25 declaring code. Oracle vigorously tried to impeach this testimony. Whether or not the practice
26 rose to the level of an entrenched custom under *Wall Data* fell by the wayside. Paragraph 27
27 of the instructions allowed the jury to consider, in evaluating good faith or not, together with
28 all other circumstances, the extent to which Google’s conduct followed or contravened any

1 recognized practice in the industry. The instructions were not adjusted to insert a further
 2 reference to custom or *Wall Data* in a second place (presumably in the concluding paragraph
 3 on the fair use).² Mentioning it twice would have elevated practice and custom to a higher
 4 profile than deserved over and above the other fair use factors. Google's point was adequately
 5 subsumed under the discussion of propriety of the accused infringer's conduct.

6 3. Deserving notice before turning to Oracle's main challenges is a tediously
 7 undramatic yet highly practical point. Our jury could reasonably have concluded as follows.
 8 Sun developed the Java programming language and made it free for all to use without a license.
 9 Sun further accumulated the copyrighted Java API library of pre-written code, including its
 10 implementing code, to carry out common and more advanced functions and made it available
 11 for all to use with a license, although the question for our jury was the extent to which, if at all,

12
 13 ² Paragraph 27 stated:

14 Also relevant to the first statutory factor is the propriety of the accused
 15 infringer's conduct because fair use presupposes good faith and fair dealing.
 16 Where, for example, the intended purpose is to supplant the copyright holder's
 17 commercially valuable right of first publication, good faith is absent. In
 18 evaluating the question of the propriety of Google's conduct, meaning good faith
 19 or not, you may only consider evidence up to the commencement of this lawsuit
 20 on August 12, 2010, and may not consider events thereafter. Your decision as to
 21 fair use, however, will govern as to all versions of Android at issue in this case,
 22 regardless of their date of issue. Again, in evaluating good faith or not, you
 23 should limit your consideration to events before August 12, 2010, and disregard
 24 any evidence you have heard after that date. This evidence cut-off date applies
 25 only to the issue of good faith or not.

26 In evaluating the extent to which Google acted in good faith or not, you may
 27 take into account, together with all other circumstances, the extent to which
 28 Google relied upon or contravened any recognized practices in the industry
 concerning re-implementation of API libraries.

You have heard evidence concerning the possibility of Google seeking a license
 from Oracle. Under the law, if the accused use is otherwise fair, then no
 permission or license need be sought or granted. Thus, seeking or being denied
 permission to use a work does not weigh against a finding of fair use.

Similarly, you have heard evidence about various licenses from the Apache
 Foundation, the Apache Harmony Project involving Java, and the General Public
 License. These are relevant in some ways, but Google concedes it had no
 license from Sun or Oracle, and it is important to remember that Google makes
 no claim that its use was pursuant to a license from Sun or Oracle, directly or
 indirectly. Instead, Google claims that its use was a fair use and therefore
 required no license at all.

1 the declaring code and its structure, sequence, and organization (“SSO”) could be carried over
2 into the Android platform without a license under the statutory right of fair use.

3 The Java API library contains, as stated, pre-written Java source code programs for
4 common and more advanced computer functions. They are organized into “packages,”
5 “classes,” and “methods.” A “package” is a collection of “classes,” and in turn, each “class”
6 is a collection of “methods” (and other elements). Each method performs a specific function,
7 sparing a programmer the need to write Java code from scratch to perform that function. Put the
8 other way, various methods are grouped under various classes with the classes grouped under
9 various packages, as in “java.lang.Math” with “java.lang” being the package and
10 “java.lang.Math” being the class. The particular taxonomy adopted for the Java API reflects
11 its unique file system, that is, its SSO.

12 Significantly, under the rules of the language itself, each method must begin with a
13 “declaration,” usually referred to herein as “declaring code.” This declares or defines (i) the
14 method name and (ii) the input(s) and their type as expected by the method and the type of
15 any outputs. After the declaration, each method next includes “implementing code,” *i.e.*, the
16 pre-written program, which takes the input(s) and, using step-by-step code, carries out the
17 function. The implementing code is set off by special punctuation.

18 A simple example of a pre-written method is one that finds a square root. At the place
19 in a developer’s own program that needs a square root (of say 81), he or she inserts a line (or a
20 “statement”) in the specified format invoking a method pre-written to find square roots. When
21 the computer runs the program and reaches this line, the computer calls upon the pre-written
22 method in its file in the Java API library, provides the method with the input (81), steps through
23 the “implementing code” to the end of the method, and finally “returns” the square root (9) to
24 the program in progress.

25 The two definitional purposes of “declaring code” are critical. The first declares the
26 precise name of the method (so that the right file will be accessed). The second specifies the
27 input(s) and their type (so that the implementing code will receive the input(s) in the way
28 expected) as well as the type of the output. In our simple square root example, there is only

one input, but it must be in parentheses after “sqrt,” which is the name of the method. In the Java API library, that particular input is a special kind of floating decimal point number (rather than a whole integer) known as a “double.” That part of the method declaration would look like this:

```
public static double sqrt(double x)
```

wherein sqrt is the method name, the input is a floating decimal number in the form of a “double” (like 81.0 or 11.56), and the method will return a “double” (like 9.0 or 3.4). (For present purposes, we may ignore the words “public” and “static.”)

In writing his or her own Java program, a programmer may only invoke a method with a statement using the precise form defined by the declaring code for the method, both as to name of method and the input format specification. To repeat, the precise name finds the precise file containing the pre-written code for that method. The precise inputs must match the format expected by the method. In our programmer’s own program, the statement calling upon the method might look like the second line just below:

```
x = 81.0
```

```
y = Math.sqrt(x)
```

wherein the right side of the statement is dictated by the declaring code and the left side y is a variable choice made by the programmer (so that y would be set to the square root, here 9.0).³

Regardless of the approach taken by the implementing code to solving the problem addressed by the method (*e.g.*, getting the square root), the input(s) to the method, to repeat, must be of the type as specified in the method declaration, so that the implementing code will receive the inputs in the type expected. Similarly, the method will return an output in the type specified in the method declaration.

Many thousands of pre-written methods have been written for Java, so many that thick books (*see, e.g.*, TX 980) are needed to explain them, organized by packages, classes, and methods. For each method, the book sets forth the precise declaring code but does not (and need not) set forth any implementing code. In other words, the book duplicates all of the

³ “Math.sqrt” corresponds to “Class.method.” The package need not be specified in our example.

1 method declarations (organized by packages and classes) together with plain English
2 explanations. A Java user can study the book and learn the exact method name and inputs
3 needed to invoke a method for use in his or her own program. The overall set of declarations is
4 called the Java Application Program Interface or Java API. Again, all that the Java programmer
5 need master are the declarations. The implementing code remains a “black box” to the
6 programmer.

7 In this important sense, the declarations are “interfaces,” meaning precise doorways
8 to command access to the pre-written methods and their implementation code performing the
9 actual work of the methods. Java users (and Android users for that matter) must invoke the
10 methods using command statements conforming to the specifications declared by the
11 declarations.

12 Oracle has portrayed the Java programming language as distinct from the Java API
13 library, insisting that only the language itself was free for all to use. Turns out, however, that in
14 order to write at all in the Java programming language, 62 classes (and some of their methods),
15 spread across three packages *within* the Java API library, *must* be used. Otherwise, the
16 language itself will fail. The 62 “necessary” classes are mixed with “unnecessary” ones in the
17 Java API library and it takes experts to comb them out. As a result, Oracle has now stipulated
18 before the jury that it was fair to use the 62 “necessary” classes given that the Java
19 programming language itself was free and open to use without a license (Tr. 1442–43; TX
20 9223).⁴

21 That the 62 “necessary” classes reside without any identification as such within the Java
22 API library (rather than reside within the programming language) supports Google’s contention
23 that the Java API library is simply an extension of the programming language itself and helps
24
25

26
27 ⁴ Java 2 SE Version 5.0 (one of the copyright works), included 166 API packages. Those packages
28 included over three thousand classes and interfaces, which, in turn, included a total of more than ten thousand
methods. Android used the declaring code and SSO of 37 of those API packages including more than six
hundred classes (and other elements) which, in turn, included more than six thousand methods. As stated, the
implementing code was not copied.

1 explain why some view the Java API declarations as free and open for use as the programming
 2 language itself. At least to the extent of the 62 “necessary” classes, Oracle agrees.⁵

3 All this said, our fair use issue, as presented to our jury, came down to whether someone
 4 using the Java programming language to build their own library of Java packages was free to
 5 duplicate, not just the “necessary” functions in the Java API library but also to duplicate any
 6 other functions in it and, in doing so, use the same interfaces, *i.e.*, declaring code, to specify the
 7 methods — so long as they supplied their own implementing code.

8 Oracle’s argument in the negative amounts to saying: Yes, all were free to use the Java
 9 programming language. Yes, all were free to use the 62 necessary classes from the Java API.
 10 Yes, all were free to duplicate the same functionality of any and all methods in the Java API
 11 library so long as they “re-implemented” (since copyright does not protect functionality or
 12 ideas, only expression). But, Oracle would say, anyone doing so should have scrambled the
 13 functionalities among a different taxonomy of packages and classes (except as to the 62
 14 “necessary” classes). That is, they should have used a different SSO.

15 Here, the undramatic yet practical point comes into sharp focus. If, as it was entitled to
 16 do, Google had simply reorganized the same functionality of the 37 re-implemented Java
 17 packages into a different SSO (taking care, however, not to disturb the 62 necessary classes and
 18 their three respective packages), then Java programmers, in order to use the Java system as well
 19 as the reorganized Android system, would have had to master and keep straight two different
 20 SSO’s as they switched between the two systems for different projects. Our jury could
 21 reasonably have found that this incompatibility would have fomented confusion and error to
 22 the detriment of *both* Java-based systems and to the detriment of Java programmers at large.
 23 By analogy, all typewriters use the same QWERTY keyboard — imagine the confusion and
 24 universal disservice if every typewriter maker had to scramble the keyboard. Since both
 25 systems presupposed the Java programming language in the first place, it was better for both to
 26

27 ⁵ Trial Exhibit 980, *The Java Application Programming Interface, Volume 1*, is a book that covers four
 28 packages and refers to them as the “core packages.” According to the back cover of the book, these four
 packages “are the foundation of the Java language. These libraries include java.lang, java.io, java.util, and
 java.net. These are the general purpose libraries fundamental to every Java program.”

1 share the same SSO insofar as they offered the same functionalities, thus maintaining usage
2 consistency across systems and avoiding cross-system confusion, just as all typewriter
3 keyboards should use the QWERTY layout — or so our jury could reasonably have found.

4 The same could have been reasonably found for the second purpose of the declaring
5 code — specifying the inputs, outputs, and their type. To the extent a specification could be
6 written in more than one way to carry out a given function, it was nevertheless better for all
7 using the Java language to master a single specification rather than having to master, for the
8 same function, different specifications, one for each system, with the attendant risk of error
9 in switching between systems — or so our jury could reasonably have found.

10 In terms of the four statutory factors, this consideration bears significantly upon the
11 nature and character of the use (the First Factor), the functional character of the declaring code
12 (the Second Factor), and the limited extent of copying (the Third Factor), that is, Google copied
13 only so much declaring code as was necessary to maintain inter-system consistency among Java
14 users. Google supplied its own code for the rest. Overall, avoiding cross-system babel
15 promoted the progress of science and useful arts — or so our jury could reasonably have found.⁶

16 This order will now turn to specific arguments raised by Oracle, the losing party, in its
17 challenge to the verdict.

18 4. With respect to Factor One, Oracle presses hard its view that Google copied
19 in bad faith disregard of Sun/Oracle's property rights. As stated, there remains an ongoing
20 debate regarding whether the "propriety of the use" is a cognizable consideration in any fair use
21 inquiry. Nevertheless, our jury was instructed, as requested by Oracle, to consider whether
22 Google acted in good faith or not as part of its consideration of the first statutory factor.
23 Although mental state is a classic question reserved to the jury, and in our trial mental state
24 was much contested, Oracle now insists that our jury could not reasonably have concluded that
25 Google acted in good faith.

26
27 ⁶ This point of inter-system consistency, by the way, differs from the interoperability point criticized
28 by the Federal Circuit. 750 F.3d at 1371. The immediate point of cross-system consistency focuses on avoiding
confusion in usage between the two systems, both of which are Java-based, not on one program written for one
system being operable on the other, the point addressed by the Federal Circuit.

Oracle cites numerous examples of internal documents and trial testimony that suggested that Google felt it needed to copy the Java API as an accelerant to bring Android to the market quicker. It points to the breakdown in negotiations between Google and Sun seeking to form a full partnership leaving Google with Java class libraries that were “half-ass at best. [It] need[ed] another half of an ass” (TX 215). In light of that breakdown, Google elected to “[d]o Java anyway and defend [its] decision, perhaps making enemies along the way” (TX 7 at 2). Oracle further notes that even after Sun’s CEO at the time publicly praised Android, Andy Rubin (head of Google’s Android team) instructed representatives at a trade show, “don’t demonstrate [Android] to any [S]un employees or lawyers” (TX 29). Finally, Oracle points to internal communications indicating that Google believed it needed a license to use Java (TX 10; *see also* TX 409 (discussing the possibility of buying Sun to “solve all these lawsuits we’re facing”)).

On the other hand, Google presented evidence that many at Google (and Sun) understood that at least the declaring code and their SSO were free to use and re-implement, both as a matter of developer practice and because the availability of independent implementations of the Java API enhanced the popularity of the Java programming language, which Sun promoted as free for all to use (Schmidt Testimony, Tr. 361; Page Testimony, Tr. 1846; Rubin Testimony, Tr. 639; Rubin Testimony, Tr. 1088–89).

Sun’s own CEO at the time, Jonathan Schwartz, testified on Google’s behalf at trial and supported Google’s view that a practice of duplicating declarations existed and that the competition was on implementations. Oracle’s harsh cross-examination focused on character assassination and showing that Schwartz resented Oracle for its treatment of Schwartz after the buyout. That Oracle resorted to such impeachment underscores how fact-bound the issue was, another classic role of a jury to resolve.

In light of the foregoing, our jury could reasonably have concluded that Google’s use of parts of the Java API as an accelerant was undertaken based on a good faith belief that at least the declaring code and SSO were free to use (which it did use), while a license was necessary for the implementing code (which it did not use). Our jury could reasonably have

1 concluded that Google's concern about making an enemy of Sun reflected concern about the
2 parties' business relationship in light of the failed negotiations that would have brought Sun
3 in as a major partner in Android, rather than concerns about litigation. Mental state was and
4 remains a classic province of the jury.

5 5. With respect to the Factor One and commercialism, it is undisputed that
6 Google's use of the declaring code and SSO from 37 Java API packages served commercial
7 purposes and our jury was so instructed, including an instruction that a commercial use weighed
8 against fair use. Nevertheless, our jury could reasonably have found that Google's decision
9 to make Android available open source and free for all to use had non-commercial purposes
10 as well (such as the general interest in sharing software innovation). Indeed, Sun itself
11 acknowledged (before Android launched) that making OpenJDK available as open source,
12 as Sun did, could undermine its own commercial efforts with Java SE licensing (TX 971 at 14).
13 Thus, even though Google's use was commercial, which weighed against fair use, the jury
14 could reasonably have found the open-source character of Android tempered Google's overall
15 commercial goals.

16 Of course, even a *wholly* commercial use may still constitute a fair use. *Campbell*,
17 510 U.S. at 585. Thus, in the alternative, our jury could reasonably have found that Google's
18 use of the declaring code and SSO from 37 Java API packages constituted a fair use despite
19 even a heavily commercial character of that use.

20 It is true that in the first appeal, the following exchange occurred at oral argument
21 between Circuit Judge Kathleen O'Malley and counsel for Google:

22 *Judge O'Malley:* But for purpose and character, though, you don't
23 dispute that it was entirely a commercial purpose.

24 *Van Nest:* No.

25 Oral Arg., *Oracle Am., Inc. v. Google Inc.*, Nos. 2013-1021, 2013-1022 (Fed. Circ.)
26 1:02:54–1:03:00.

27 On remand, Oracle sought to convert this colloquy to a judicial admission that Google's
28 use was "entirely commercial." It is for the district court, in its discretion, to determine the
extent, if any, of a judicial admission. *American Title Ins. Co. v. Lacelaw Corp.*, 861 F.2d 224,

226 (9th Cir. 1988). As set forth in the final pretrial order (Dkt. No. 1760), the undersigned examined the colloquy (and all other statements of record on the point) and determined that the “commercial” part would be treated as a judicial admission, but the “entirely” part would not be. The word “entirely” was part of the give and take of an oral argument. In light of all statements by counsel and in light of the free and open availability of Android, the word “entirely” would have been too conclusive, inaccurate, and unfair. The district court exercised its discretion to limit the admission to “commercial” and let the jury decide for itself how commercial, according to the evidence.

Accordingly, our jury was instructed that Google’s use was commercial, but that it was up to the jury to determine the extent of the commerciality, as follows (Dkt. No. 1981 ¶ 21) (emphasis added):

In evaluating the first statutory factor, the extent of the commercial nature of the accused use must be considered. *In this case, all agree that Google’s accused use was commercial in nature but disagree over the extent.* Commercial use weighs against a finding of fair use, but even a commercial use may be found (or not found, as the case may be) to be sufficiently transformative that the first statutory factor, on balance, still cuts in favor of fair use. To put it differently, the more transformative an accused work, the more other factors, such as commercialism, will recede in importance. By contrast, the less transformative the accused work, the more other factors like commercialism will dominate.

Our jury could reasonably have agreed with Oracle that the evidence showed the use was entirely commercial (yet still ruled for Google), but it could also have reasonably found that the use, while commercial, served non-commercial purposes as well, *i.e.*, as part of a free and open software platform, namely Android.⁷

6. With respect to the Factor One and “transformativeness,” a use is transformative if it “adds something new, with a further purpose or different character, altering the first with new expression, meaning, or message.” *Campbell v. Acuff-Rose Music, Inc.*, 510 U.S. 569, 579

⁷ Although Google gives Android away for free, Oracle argues that the “Android Ecosystem” has generated over forty billion dollars in revenue and thus Android has had a massive commercial benefit to Google. There is no doubt that Android has contributed to a large expansion of smartphones but the revenue benefit to Google flows from the ad revenue generated by its search engine which pre existed Android. In other words, our jury could reasonably have found that without Android the void would have been filled by other mobile platforms, yet those platforms would still have led to more Google search requests and ad revenue.

(1994). Oracle argues that no jury could reasonably find that Google’s use of the declaring code and SSO from 37 Java API packages in Android imbued the copyrighted works with new expression, meaning, or message. Specifically, Oracle argues that the copied code served the same function in Android as it did in Java, inasmuch as the code served as an interface for accessing methods in both systems (*see* Astrachan Testimony, Tr. 1265; Bloch Testimony, Tr. 997).

It should go without saying (but it must be said anyway) that, of course, the words copied will always be the same (or virtually so) in a copyright case — otherwise there can be no copyright problem in the first place. And, of course, the copied declarations serve the same function in both works, for by definition, declaring code in the Java programming language serves the specific definitional purposes explained above. If this were enough to defeat fair use, it would be impossible ever to duplicate declaring code as fair use and presumably the Federal Circuit would have disallowed this factor on the first appeal rather than remanding for a jury trial.

With respect to transformativeness, our jury could reasonably have found that (i) Google’s selection of 37 out of 166 Java API packages (ii) re-implemented with new implementing code adapted to the constrained operating environment of mobile smartphone devices with small batteries, and (iii) combined with brand new methods, classes, and packages written by Google for the mobile smartphone platform — all constituted a fresh context giving new expression, meaning, or message to the duplicated code.⁸ (The copyrighted works were designed and used for desktop and laptop computers.)

In *Campbell*, the accused work (a rap parody song) used the same bass riff and an identical first line of Roy Orbison’s “Oh, Pretty Woman.” The parody also included exact copies of certain phrases in subsequent lines and maintained the same structure and rhyme

⁸ As stated, the Android core libraries included over one hundred new API packages that had never been part of the Java API. Those packages enabled functionality specifically intended for use in a mobile smartphone environment, and like the 37 Java API packages at issue here, they were written in the Java programming language (Rubin Testimony, Tr. 670). Some additional functionality in Android, however, was performed by a separate set of libraries written in C or C++ for performance purposes (Douglas Schmidt Testimony, Tr. 1602).

1 scheme throughout. The copied elements served the same function in the accused work as in
 2 the original. Nevertheless, the Supreme Court acknowledged that the transformative purpose
 3 of parody had a “need to mimic an original to make its point,” and thus, warranted copying some
 4 exact elements. *Id.* at 580–81. The question of the *extent* of the copying permissible to serve
 5 that function was the subject of the inquiry of the third statutory fair use factor. So too here.

6 Android did not merely incorporate the copyrighted work “as part of a broader work,”
 7 without any change to the purpose, message, or meaning of the underlying work (*see* Dkt.
 8 No. 1780). Android did not merely adopt the Java platform wholesale as part of a broader
 9 software platform without any changes. Instead, it integrated selected elements, namely
 10 declarations from 37 packages to interface with all new implementing code optimized for mobile
 11 smartphones and added entirely new Java packages written by Google itself. This enabled a
 12 purpose distinct from the desktop purpose of the copyrighted works — or so our jury could
 13 reasonably have found.

14 In light of the foregoing, our jury could reasonably have concluded that Google’s use of
 15 the declaring code and SSO of 37 API packages from the desktop platform work in a full-stack,
 16 open-source mobile operating system for smartphones was transformative.⁹

17 7. With respect to Factor Two, the “nature of the copyrighted work,” the final
 18 charge to the jury stated “[t]his factor recognizes that traditional literary works are closer than
 19 informational works, such as instruction manuals, to the core of intended copyright protection.
 20 Creative writing and expression lie at the very heart of copyright protection, so fair use is

21
 22
 23
 24
 25 ⁹ The instructions on “transformativeness” deleted a point from the Federal Circuit opinion that might
 26 have favored Google, which had requested an instruction defining “transformative” as the incorporation of
 27 copyrighted material “as part of a broader work,” relying on a parenthetical snippet in the Federal Circuit
 28 opinion. This Court denied Google’s request and explained why (Dkt. 1780). In brief, the parenthetical snippet
 was taken from our court of appeal’s decision in *Monge v. Maya Magazines*, 688 F.3d 1164, 1176 (9th Cir.
 2012). But as our court of appeals there explained, the incorporation of a copyrighted material into a larger
 work, such as the arrangement of a work in a photo montage, *could be* transformative and fair use, not that it
must be. Please see the order at Docket Number 1780 for the reasoning.

generally more difficult to establish for copying of traditional literary works than for copying of informational works” (Dkt. No. 1981 ¶ 28); *see also Campbell*, 510 U.S. at 586.¹⁰

The Java programming language itself requires the package-class-method hierarchy, an idea on which Oracle does not claim any copyright. Oracle instead argues that because there were countless ways to name and organize the packages in Java and because Google could have used a completely new taxonomy in Android (except as to the 62 “necessary” classes), our jury should have concluded that the process of designing APIs must have been “highly creative” and thus at the core of copyright’s protection. Of course, such a conclusion would have been within the evidence, but our jury could reasonably have gone the other way and concluded that the declaring code was not highly creative.

Oracle highlights Google’s own witness, Joshua Bloch, who designed many of the Java APIs while working at Sun and who later worked at Google on the Android team. Bloch testified that one of the challenges he faced in designing API was “the complexity of figuring out how best to express what it is that the programmer wants done” (Tr. 1007). Oracle focuses on Bloch’s use of the word “express” to demonstrate the expressive nature of API design but it ignores the fact that he addressed the challenge of expressing a particular *function*. Similarly, Oracle notes that Bloch described API design as “an art not a science” and cites his eloquence regarding “design principles” (Tr. 971).

In citing this, Oracle resorts to the time-honored tactic of emphasizing a concession by one of the other side’s witness. But other witnesses (*e.g.*, Dr. Owen Astrachan, among others) emphasized the functional role of the declaring lines of code and their SSO and minimized the “creative” aspect. Our jury could reasonably have found that, while the declaring code and SSO were creative enough to qualify for copyright protection, functional considerations predominated in their design, and thus Factor Two was not a strong factor in favor of Oracle after all.

¹⁰ “[I]f a work is largely functional, it receives only weak protection. ‘This result is neither unfair nor unfortunate. It is the means by which copyright advances the progress of science and art.’” *Sega Enterprises, Ltd. v. Accolade, Inc.*, 977 F.2d 1510, 1527 (9th Cir. 1992) (quoting *Feist Publications, Inc. v. Rural Tel. Serv. Co., Inc.*, 499 U.S. 340, 350 (1991)).

1 8. With respect to Factor Three, our jury could reasonably have found that Google
2 duplicated the bare minimum of the 37 API packages, just enough to preserve inter-system
3 consistency in usage, namely the declarations and their SSO only, and did not copy any of the
4 implementing code, thus finding that Google copied only so much as was reasonably necessary
5 for a transformative use. The number of lines of code duplicated constituted a tiny fraction of
6 one percent of the copyrighted works (and even less of Android, for that matter).

7 9. With respect to Factor Four, our jury could reasonably have found that use of the
8 declaring lines of code (including their SSO) in Android caused no harm to the market for the
9 copyrighted works, which were for desktop and laptop computers. As to Java ME, our jury
10 could reasonably have found that Java ME eventually declined in revenue just as predicted by
11 Sun before Android was even released, meaning that Android had no further negative impact on
12 Java ME beyond the tailspin already predicted within Sun.

13 Also, before Android was released, Sun made all of the Java API available as free and
14 open source under the name OpenJDK, subject only to the lax terms of the General Public
15 License Version 2 with Classpath Exception. This invited anyone to subset the API.
16 Anyone could have duplicated, for commercial purposes, the very same 37 packages as wound
17 up in Android with the very same SSO and done so without any fee, subject only to lenient
18 “give-back” conditions of the GPLv2+CE. Although Google didn’t acquire the 37 packages
19 via OpenJDK, our jury could reasonably have found that Android’s impact on the market for the
20 copyrighted works paralleled what Sun already expected via its OpenJDK.

21 10. Stepping back, it seems hard to reconcile Oracle’s current position with the one it
22 took just as the trial was getting underway, namely, that fair use is an equitable rule of reason
23 and each case requires its own balancing of factors. In its critique of the first proposed jury
24 instructions on fair use (Dkt. No. 1663 at 1), Oracle argued that the Court’s draft characterization
25 of the policy of fair use contravened the legislative history, and Oracle cited the following
26 language from a Senate report on the 1976 Copyright Act (which language was repeated in the
27 House Report):

28 Although the courts have considered and ruled upon the fair use
 doctrine over and over again, no real definition of the concept has

1 ever emerged. Indeed, since the doctrine is an equitable rule of
2 reason, no generally applicable definition is possible, and each
 case raising the question must be decided on its own facts.

3 S.Rep. No. 94-473 at 62 (1975). The Court adopted Oracle’s proposed instruction in the next
4 draft as well as in the final charge to the jury, stating: “Since the doctrine of fair use is an
5 equitable rule of reason, no generally accepted definition is possible, and each case raising the
6 question must be decided on its own facts” (Dkt. No. 1981 ¶ 21).

7 Now, Oracle argues instead that this case must be decided as a matter of law, and not
8 “on its own facts.” Oracle argues that Google’s copying fails to resemble any of the statutory
9 examples of fair use listed in the precatory language of Section 107, again contradicting its
10 earlier position that “no generally applicable definition is possible.”

11 In applying an “equitable rule of reason,” our jury could reasonably have given weight
12 to the fact that cross-system confusion would have resulted had Google scrambled the SSO
13 and specifications. Java programmers and science and the useful arts were better served by
14 a common set of command-type statements, just as all typists are better served by a common
15 QWERTY keyboard.

16 11. In summary, on Factor One, our jury could reasonably have found that while the
17 use was commercial, the commercial use was outweighed by a transformative use, namely use
18 of the declaring code as one component in a full stack platform for highly advanced
19 smartphones, a different context in which (i) 37 of the 166 API packages were selected, (ii) all of
20 the implementing code was re-implemented for a mobile low-power platform, and (iii) many
21 new packages original with Android were added. Despite Google’s internal e-mails, our jury
22 could reasonably have found that most of them pertained to earlier negotiations for a joint
23 venture to use the *entire* Java system, including the implementing code, and that, after those
24 discussions failed, Google acted in good faith by duplicating only the declarations to 37
25 packages to maintain inter-system consistency in usage and by supplying its own implementing
26 code. On Factor Two, our jury could reasonably have found that the code copied was not highly
27 creative, was mainly functional, and was less deserving of protection. On Factor Three, our jury
28 could reasonably have found that Google duplicated only the declaring code, a tiny fraction of

1 the copyrighted works, duplicated to avoid confusion among Java programmers as between the
2 Java system and the Android system. On Factor Four, our jury could have found that Android
3 caused no harm to the desktop market for the copyrighted works or to any mobile derivative, as
4 borne out by Sun's own records. Of course, Oracle had arguments going the other way, but the
5 jury was reasonably within the record in finding fair use.

6 This order cannot cover all the myriad ways that the jury could reasonably have balanced
7 the statutory factors and found in favor of fair use. The possibilities above represent but one
8 take on the evidence. Witness credibility was much challenged. Plainly, many more variations
9 and balancings could have reasonably led to the same verdict.

10 12. A final word about a separate issue that arose during trial. In their joint final
11 pretrial submission, both sides agreed that no reference would be made before the jury to the
12 prior proceedings in this case (Dkt. No. 1709 at 8). As this trial developed, however, Oracle left
13 the impression before the jury that all the way up to the present, Google had uniformly acted in
14 bad faith. Problem was, during a substantial part of this period (2012–2014), Google had been
15 entitled to rely on the judgment of the district court that the material asserted was not
16 copyrightable. *Kamar Int'l, Inc. v. Russ Berrie & Co.*, 752 F.2d 1326, 1330 (column two)
17 (9th Cir. 1984) stated (emphasis added):

18 We affirm the district court's holding that the sales by Russ Berrie
19 of its stuffed animals immediately following the first judgment do
20 not count as infringements after notice. *Kamar's* supposed citation
21 to the contrary . . . is wholly inapposite. In its first judgment, the
district court held Russ Berrie's animals noninfringing. *Kamar* did
not obtain any stay pending appeal. *Russ was entitled to rely on*
the judgment at that time.

22 In response, Oracle contended that Judge Alex Kozinski's opinion for our court of
23 appeals in *Micro Star v. Formgen, Inc.*, 154 F.3d 1107 (9th Cir. 1998), had been so at odds
24 with the decision by this Court holding that the declaring code and their structure, sequence
25 and organization were not copyrightable that Google could not reasonably have believed that
26 this Court's holding on uncopyrightability was correct (Trial Tr. at 1591). The short answer
27 was that *Micro Star* provided no holding or dictum whatsoever on copyrightability — none.
28 Copyrightability was not there raised. (It was a fair use case.) Indeed, in our earlier trial when

copyrightability was debated, no one, including Oracle, ever cited *Micro Star* on copyrightability. Nor was it raised on appeal.

To resolve this problem of the 2012-2014 interregnum period as best as could be done with minimal strain on the parties' stipulation, the Court gave the following instruction:

In evaluating the question of the propriety of Google's conduct, meaning good faith or not, you may only consider evidence up to the commencement of this lawsuit on August 12, 2010, and may not consider events thereafter. Your decision as to fair use, however, will govern as to all versions of Android at issue in this case, regardless of their date of issue. Again, in evaluating good faith or not, you should limit your consideration to events before August 12, 2010, and disregard any evidence you have heard after that date. This evidence cut-off date applies only to the issue of good faith or not.

No mention was made to the jury about the earlier judgment rejecting copyrightability. The problem was largely solved by the date cut-off, which allowed Oracle to use all of Google's "bad" e-mails. To mitigate the problem of speculation regarding prior testimony read in at the second trial, the following instruction was given:

You may have heard from a witness that there was a prior trial in this case. It is true that there was a prior trial. We have heard evidence in this trial of a prior proceeding, which is the earlier trial that occurred in this case. Do not speculate about what happened in the prior trial. No determination on fair use was made one way or the other in that trial. It is up to you, the jury, to determine fair use based on the evidence you have heard in this trial and my instructions of the law.


Unfortunately, this might not have eliminated all of the prejudice to Google from the suggestion made before the jury by Oracle, but it went most of the way and was the best the Court could do in light of the stipulation made by the parties at the outset.

* * *

All Rule 50 motions are **DENIED**. Judgment will be entered in accordance with the jury's verdict.

IT IS SO ORDERED.

Dated: June 8, 2016.



WILLIAM ALSUP
UNITED STATES DISTRICT JUDGE